e35

PCT

WORLD INTELLECTU
Inten

INTERNATIONAL APPLICATION PUBLISHED

WO 9603692A1

(51) International Patent Classification 6 :
G06F 9/46

A1

(43) International Publication Date: 8 February 1996 (08.02.96)

(21) International Application Number: PCT/GB95/01705

(22) International Filing Date: 19 July 1995 (19.07.95)

(30) Priority Data:
9414951.5 25 July 1994 (25.07.94) GB

(71) Applicant (*for all designated States except US*): BRITISH TELECOMMUNICATIONS PUBLIC LIMITED COMPANY [GB/GB]; 81 Newgate Street, London EC1A 7AJ (GB).

(72) Inventors; and
(75) Inventors/Applicants (*for US only*): BUTT, John [GB/GB]; 212 Ashcroft Road, Ipswich, Suffolk IP1 6AF (GB). IRELAND, Paul, Stuart [GB/GB]; 28 Richardsons Road, East Beryholt, Colchester, Essex C07 6RR (GB).

(74) Agent: EVERSHED, Michael; BT Group Legal Services, Intellectual Property Dept., 13th floor, 151 Gower Street, London WC1E 6BA (GB).

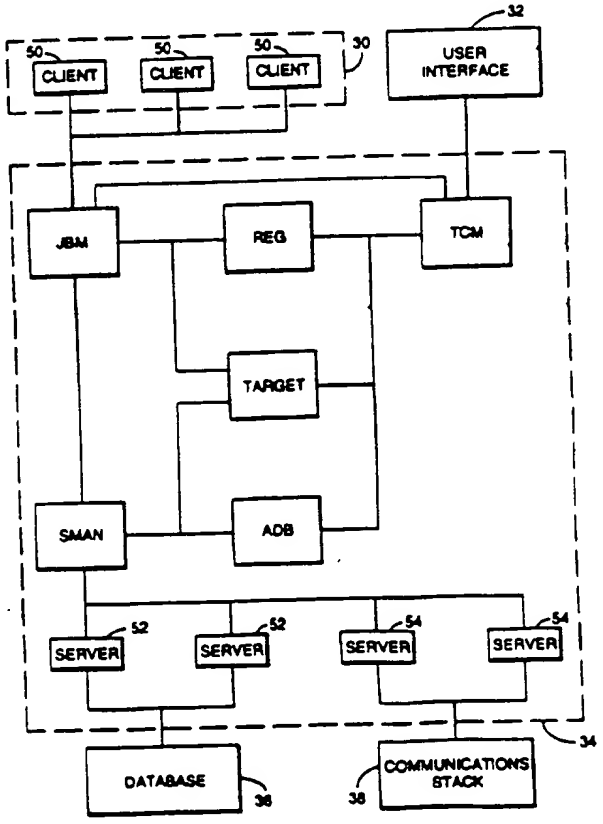
(81) Designated States: AM, AU, BB, BG, BR, BY, CA, CN, CZ, EE, FI, GB, GE, HU, IS, JP, KE, KG, KP, KR, KZ, LK, LR, LT, LV, MD, MG, MN, MW, MX, NO, NZ, PL, RO, RU, SD, SG, SI, SK, TJ, TM, TT, UA, UG, US, UZ, VN, European patent (AT, BE, CH, DE, DK, ES, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, ML, MR, NE, SN, TD, TG), ARIPO patent (KE, MW, SD, SZ, UG).

Published
With international search report.

(54) Title: COMPUTER SYSTEM HAVING CLIENT-SERVER ARCHITECTURE

(57) Abstract

A network manager for a telecommunications network has a client-server architecture. The software components of the network manager include a set of clients (50) which form part of the application programs of the network manager, a user interface (32), a database (36) containing details of the network and a communications stack (38) for communicating with exchanges managed by the network manager. The clients (50) generate requests to run jobs on servers (52) and (54). The jobs which are run on server (52) are eventually destined for a resource in the form of a database (36), while the jobs which are run on servers (54) are eventually destined for resources in the form of exchanges. The requests are initially passed to software module JBM. This module then checks if the resource for which the job is destined is free and, if not, puts the job on a holding queue. If the resource is free, it checks whether the job is scheduled for immediate execution or execution at a future time. If it is scheduled for execution at a future time, it is put on a queue of scheduled jobs. If the job is scheduled for immediate execution, the module JBM checks if a server is free to accept the job. If no server is free, the job is put on a queue of jobs awaiting immediate execution. If the server is free, the module JBM instructs a module SMAN to load the job onto the free server.



FOR THE PURPOSES OF INFORMATION ONLY

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

AT	Austria	GB	United Kingdom	MR	Mauritania
AU	Australia	GE	Georgia	MW	Malawi
BB	Barbados	GN	Guinea	NE	Niger
BE	Belgium	GR	Greece	NL	Netherlands
BF	Burkina Faso	HU	Hungary	NO	Norway
BG	Bulgaria	IE	Ireland	NZ	New Zealand
BJ	Benin	IT	Italy	PL	Poland
BR	Brazil	JP	Japan	PT	Portugal
BY	Belarus	KE	Kenya	RO	Romania
CA	Canada	KG	Kyrgyzstan	RU	Russian Federation
CF	Central African Republic	KP	Democratic People's Republic of Korea	SD	Sudan
CG	Congo			SE	Sweden
CH	Switzerland	KR	Republic of Korea	SI	Slovenia
CI	Côte d'Ivoire	KZ	Kazakhstan	SK	Slovakia
CM	Cameroon	LI	Liechtenstein	SN	Senegal
CN	China	LK	Sri Lanka	TD	Chad
CS	Czechoslovakia	LU	Luxembourg	TG	Togo
CZ	Czech Republic	LV	Latvia	TJ	Tajikistan
DE	Germany	MC	Monaco	TT	Trinidad and Tobago
DK	Denmark	MD	Republic of Moldova	UA	Ukraine
ES	Spain	MG	Madagascar	US	United States of America
FI	Finland	ML	Mali	UZ	Uzbekistan
FR	France	MN	Mongolia	VN	Viet Nam
GA	Gabon				

COMPUTER SYSTEM HAVING CLIENT-SERVER ARCHITECTURE

This invention relates to a computer system having a client-server architecture and also to a method of operating a computer system having this architecture.

A computer system having a client-server architecture comprises a set of software modules known as clients and a further set of software modules, known as servers, for serving requests from the clients to run jobs. This type of architecture enables jobs from the clients to be distributed among the servers. The servers may be of similar or differing types. With this type of architecture, it is desirable to provide a mechanism for controlling the scheduling of requests by the clients to run jobs on the servers.

According to one aspect of this invention there is provided a computer system having a client-server architecture, said system comprising: a set of clients; a set of servers for serving requests from the clients to run jobs; means for managing requests from the clients to run jobs on servers; and means for loading jobs on to servers; said request managing means being arranged to receive requests from the clients and, on receiving a request, to perform the following operations: check if a server is free to run the job; if a server is not free, put the job on a queue for jobs each of which is ready for execution when a server becomes free; and when a server is free to run a job, instructing said job loading means to load a job on to the server.

The provision of a queue for jobs each of which is ready for execution when a server becomes free enables the computer system to deal with the situation where clients are requesting jobs to be run on servers at a time when a server is not free to run the jobs.

According to a second aspect of this invention there is provided a computer system having a client-server architecture, said computer system comprising: a set of clients; a set of servers for serving requests from the

clients to run jobs; means for managing requests from clients to run jobs on servers; and means for loading jobs on to servers; said request managing means being arranged to receive requests from the clients and, on receiving a request from a client to run a job which is destined for a resource accessed by a server, to perform the following operations; check is the resource is free to run the job; and if the resource is not free, put the job on a queue for jobs each of which is waiting for a resource to become free.

According to a third aspect of this invention there is provided a method of operating a computer system having client-server architecture, said computer system comprising a set of clients and a set of servers for serving requests from the clients to run jobs, for each request made by a client to run a job on a server, said method comprising the steps of:

checking if a server is free to run the job; and

if a server is not free to run the job, putting the job on a queue for jobs each of which is ready for execution when a server becomes and

when a server is free to run the job, loading a job to the server.

According to a fourth aspect of this invention there is provided a method of operating a computer system having client-server architecture, said computer system having a set of clients and a set of servers for serving requests from the clients to run jobs; for each request made by a client to run a job on a resource accessed by a server, said method including the steps of: checking if the resource is free to run the job; and, if the resource is not free, putting the job on a queue for jobs each of which is waiting for resource to become free.

This invention will now be described in more detail by way of example, with reference to the drawings in which

Figure 1 is a block diagram of a network manager and associated element managers and local exchanges, the network manager including a computer system embodying this invention

Figure 2 is a block diagram of the main software components of the network manager of Figure 1;

Figure 3 is a block diagram showing the individual modules which form the transaction processing component of the network manager of Figure 1 together with its relationship to the other software components;

each of Figures 4 to 6 is a flow chart illustrating the operation of one of the modules of the transaction processing component;

Figure 7 is a block diagram of set of computers which together form a computer system having a client-server architecture; and

Figure 8 is a block diagram of the hardware components of the network manager shown in Figure 1.

Referring now to Figure 1, there are shown three local exchanges 10, 12, 14 which form part of a public telecommunications network. Although not shown in Figure 1, the local exchanges are connected to trunk exchanges and the trunk exchanges are all fully interconnected to each other. The local exchanges 10, 12, 14 are managed, respectively, by element managers 16, 18, 20. The three element managers 16, 18, 20 are managed by a network manager 22. Although not shown in Figure 1, each of the remaining exchanges of the network is managed by a respective individual element manager. The element managers are managed by further network managers, not shown.

Because of the complexity of an exchange, each exchange is provided with an individual element manager. By way of modification, the exchanges may be managed directly by the network manager 22 without the intermediate use of element managers. Element managers are also provided for the other elements of the network, such as multiplexers, and each of these element managers manages typically many individual elements. These further element managers are managed by one or more further network managers, not shown.

The network manager 22 sends instructions to the element managers for configuring the exchanges managed by

them. The instructions sent to each element manager ca include instructions to connect or disconnect customers fro the local exchange as well as instructions for handling call by that exchange. The element managers also receive dat from the exchanges which they manage regarding thei operating state and pass this data to the network manager 22 The network manager 22 has a database in which is stored dat relating to the state of the exchanges.

The general arrangement of network manager 22, elemen managers and exchanges managed by the element managers a described above with reference to Figure 1 is well known t those skilled in the art. By way of example, the loca exchanges 10, 12, 14 may be System X exchanges manufacture by GEC Plessey Telecommunications plc. The network manage 22 is implemented as a computer. The main hardware component of the computer which implements the network manager 22 ar shown in Figure 8. These comprise storage 60, a centra processing unit (CPU) 62, a visual display unit (VDU) 64, keyboard 66 and input/output ports 68. The storage 6 comprises hard disc memory, random-access-memory (RAM) ar read-only-memory (ROM). The software which controls th computer is stored in storage 60. The software includes client-server architecture embodying this invention. Th software of the network manager 22 will now be described i more detail with special reference to the client-serve architecture.

Referring now to Figure 2, there are shown the mai software components of the network manager 22. Thes comprise a set of application programs 30, a user interfac 32, a transaction processing component 34, a database 36 a a communications stack 38. Although not shown, t components also include the operating system for the comput which implements the network manager 22.

The application programs 30 are the programs which a responsible for sending instructions to the element manage and receiving data from them. The construction of su programs for a network manager is generally well known

those skilled in the art. The user interface 32 is the software component which permits the user to access the network manager and the construction of user interfaces is also generally well known to those skilled in the art.

The database 36 is the database mentioned above which contains data relating to the operational state of the exchanges. By way of example, the database 36 may be the well known ORACLE database management system. The communications stack 38 is responsible for converting both outgoing and incoming messages between the form used by the network manager 22 and the form which is suitable for transmission along the communication links which connect the network manager 22 and the various element managers. The construction of communications stacks is generally well known to those in the art and it is now usual for a communications stack to be provided as a standard component of the operating system of a computer.

The application programs 30 generate requests for both the database 36 and the communications stack 38 to perform jobs. In the case of the database 36, the jobs take the form of either requests to enter data into the database 36 or to retrieve data from it. In the case of the communications stack 38, the jobs take the form of requests to send messages to the element managers. The transaction processing component 34 is responsible for scheduling the jobs and this component will now be described in more detail with reference to Figure 3.

Referring now to Figure 3, there are shown the individual software modules which form the transaction processing component 34 together with the relationship of these modules to the other software components of the network manager 22. These other components comprise the user interface 32, the database 36, the communications stack 38 and a set of client modules 50. In Figure 3, for reasons of simplicity, there are shown only three client modules 50 but, in practice, there would be a much larger number of these modules. In the present example, each of the client modules

50 is one of the application programs 30. By way of modification, the client modules could form part of the transaction processing component 34 and serve the function of interfacing to the individual application programs.

The transaction processing component 34 comprises software modules JBM, SMAN, REG, TARGET, ADB and TCM. As shown in Figure 3, the transaction processing component 34 also has two servers 52 for accessing the database 36 and two servers 54 for accessing the communications stack 38. Thus, the servers 52 provide an interface to the database 36 and the servers 54 provide an interface to the communications stack 38. Thus, a request to run a job on one of the servers 52 also represents a request to run the job on the database 36. The communications stack 38 is part of an interface to the element managers 16, 18, 20 and, as will be recalled these element managers manage the local exchanges 10, 12, 14. Thus, a request to run a job on one of the servers 52 represents, ultimately, a request to run the job on one of the local exchanges. Thus, database 36 and the local exchanges 10, 12, 14 represent resources accessed by the servers. The servers 52 belong to a first type of server and the servers 54 belong to a second type of server. For reasons of simplicity, only two servers are shown of each type but, in practice, there will be more than two servers of each type.

The general function of each of the software modules which form the transaction process in component 34 will now be outlined and this will be followed by a detailed description of each of these modules.

The module JBM is responsible for the management of each request received from one of the clients 50. The module JBM checks each request to see if the job can be executed immediately. If it cannot be executed immediately, it is placed on a queue. If a job can be executed immediately, it is passed to the module SMAN which loads it on to a server. The module REG keeps a list of clients which have registered with the transaction processing component 34 and from which

requests can be accepted. The module TARGET keeps account of the number of jobs running on each of the resources accessed by the servers 52, 54. The module ADB maintains details of each executed job. The module TCM provides the user with access to the modules JBM, REG, TARGET and ADB

In more detail, the module JBM receives the requests to run jobs from the servers 50. The steps for processing each job request in the module JBM will now be described with reference to Figure 4.

On receiving a request to run a job from a client, in a step S10 a check is made with the module REG to determine if the client is registered with the transaction processing component 34. The purpose of this step is to prevent jobs being run which are received by accident from clients which are not registered. If the client is not registered, processing of the job is terminated. If the client is registered, in a step S11, a check is made with the module TARGET to determine if the resource for which the job is destined is free. If the resource is not free, in a step S12, the job is placed on a queue (the holding queue) of jobs which are waiting for a resource to become free.

Some jobs are scheduled for execution at a later time. If it is found in step S11 that the resource is free, in a step S12 a check is made to determine if the job is scheduled for execution at a later time. If the job is scheduled for execution at a later time, in a step S13 it is placed on a queue (the queue of scheduled jobs) for jobs which are scheduled for execution at a later time.

If in step S12 it is found that the job is ready for immediate execution, in a step S13 a check is made with the module SMAN to determine if a server is free to run the job. For each of the two types of server, the module JBM has a queue (a ready queue) of jobs which are waiting for a server to become free. If in step S13 it is found that a server is not free to run the job, then in a step S14 the job is placed on an appropriate one of the two ready queues. If in step S13 it is found that a server is free to run the job, in a

step S15, the module JBM instructs the module SMAN to loa the job on to the server.

As will be explained below with reference to t module SMAN, when a job has been run on a server, it unloaded from the server by the module SMAN and this modu then informs the module JBM. The module JBM then de-queu a job from the appropriate ready queue in accordance with predefined criterion and instructs the module SMAN to load on to the free server. The module SMAN has four predefin criteria for selecting the next job to be loaded onto a fr server. By using the module TCM, the user can select, f each type of server, the particular predefined criterion be used. These four predefined criteria will now described.

The first predefined criterion is simply that the ne job in the appropriate ready queue is selected as the ne job to be loaded onto the free server.

The second predefined criterion is that the next j in the appropriate ready queue for the same resource as previous job is selected as the next job to be loaded o the free server. Thus, if the previous job was destined local exchange 10, the module JBM selects the next job in appropriate ready queue for the local exchange 10 as the n job to be loaded onto the free server. If there is no job the appropriate ready queue for the same resource as previous job, the next job in the ready queue is selected the new job.

The third criterion is that the next job of the sa type, regardless of the resource for which it is destined, the appropriate ready queue is selected as the next job to loaded onto the free server. Thus, if the previous job to connect a new customer to a local exchange, the module selects the next job for connecting a new customer to a lo exchange as the next job to be loaded onto the free ser If there is no job in the ready queue of the same type as previous job, the next job in the ready queue is selecte the next job to be loaded onto the free server.

The fourth predefined criterion is that the next job in the appropriate ready queue of the same type and destined for the same resource as the previous job is selected as the next job to be loaded onto the free server. If there is no job in the ready queue of the same type and destined for the same resource as the previous job, then the next job which is destined for the same resource as the previous job is selected as the next job to be loaded onto the free server. If there is no job in the ready queue destined for the same resource as the previous job, then the next job in the ready queue is selected as the next job to be loaded onto the free server.

In the queue of scheduled jobs, the jobs are arranged in order by their scheduled times of execution. For the job which is scheduled to be executed next, a timer is established. At the time at which the job should be scheduled, further processing of the job commences at step S13.

As will also be explained below, when a resource becomes free, the module TARGET informs the module JBM. The module JBM then removes the next job from the holding queue which is destined for the resource which has become free and processing of this job recommences with step S12.

The module JBM receives requests to run two types of jobs. In the first type of jobs (attached jobs), the client and server are connected together while the job is being run on the server. In the second type of jobs (detached jobs), the client and server are not connected while the job is run. Detached jobs have the advantage that they can be run without occupying the client. When the module JBM receives a request to run a detached job, the details of the job are stored and then retrieved when the job is unloaded from a server.

Referring now to Figure 5, there are shown the steps which are performed by the module SMAN on receiving a request from the module JBM to load a job on to a server. After receiving a request to load a job on to a server, in a step S20, the module SMAN loads the job on to a free server.

Then, in a step S21, it waits for notification from the server that the job has been completed. When it receives notification that the job has been completed, in a step S22, it unloads a job from the server. In the case of an attached job, the job is unloaded by breaking the connection between the client and the server. In the case of a detached job the job is unloaded by transmitting the results of the job to the client from which the request came. Then, in a step S23 the module SMAN notifies the module JBM that the job has been completed and that the server has become free. In a step S24, the module SMAN notifies the module TARGET that the job has been completed. The module TARGET uses this data to keep count of the number of jobs which are being run on the relevant resource. Finally, in a step S25, the module SMAN notifies the module ADB that the job has been completed together with the details of the job. The module ADB uses this data to provide a log of completed jobs.

The module SMAN is also arranged to create and delete servers. For each type of server, the module SMAN maintains a table of the existing servers of that type together with a table containing the number jobs waiting in the ready queue for execution on that type of server. The module SMAN is arranged to maintain the number of servers of each type at an optimum level for the number of jobs which are awaiting execution. The procedure for achieving this for each ready queue in shown in Figure 6. In a step S30, the number of jobs on the ready queue is compared with the number of servers for serving jobs on that queue. Using a criterion pre-set by the user, the comparison is used to establish if servers should be created or deleted so as to achieve the optimum number of servers. For example, the criterion could be that the ratio of the number of jobs in the ready queue to the number of servers should be kept at a certain value. Then, in a step S31, servers are created or deleted as appropriate.

As will be explained below, the user is able to instruct the module SMAN to create and delete servers.

The module REG maintains a list of clients which are registered with the transaction processing component 34. By using the module TCM, the user can add and delete clients and inspect the list. As explained above, the list of clients is accessed by the module JBM each time it receives a request to run a job from one of the clients.

For any resource, the number of jobs which can be run at any one time efficiently is limited. The database 36 is able to handle a relatively large number of jobs simultaneously. In contrast, as the main function of the local exchanges 12, 12, 14 is to process telecommunication calls, the amount of computing capacity to run jobs requested by the network manager 22 is limited and only a few of such jobs can be run simultaneously.

The function of the module TARGET is to keep a count of the number of jobs which are running on each resource and also to maintain a threshold value for the maximum number of jobs which can be run on each resource. For each resource the module TARGET maintains a list of jobs which are running on it. When the module SMAN unloads a job from a server, it notifies the module TARGET which then deletes the job from the list for the appropriate resource. As explained above, when the module JBM receives a request to run a new job, it checks with the module TARGET if the resource is free to run the job. The module TARGET then compares the number of jobs running on that resource with its threshold value and thus determines if the resource is free to accept a new job. If the resource is free to accept a new job, the module TARGET adds the job to the list for that resource and informs the module JBM that the resource can accept a new job. If the resource is not free to accept a new job, the module TARGET informs the module JBM of this.

By using the module TCM, the user can change the threshold value for each resource and also obtain a list of jobs presently running on each resource.

The module ADB maintains a database containing details of completed jobs. Each time a job is unloaded from a server

the module SMAN sends details of the completed job to th module ADB and the module ADB stores these details in it database. The module ADB also maintains files containin details of jobs. At the end of each day, details of the job are transferred from the database to the files following selection procedure established by the user. Thus, th database requires only limited amount of data storag capacity. By using the module TCM, the user can inspect th data in the database and in the files. Thus, the module AD enables the user to monitor the performance of th transaction processing component 34.

The module TCM provides the user with access to th modules JBM, REG, TARGET and ADB. In the case of the modul JBM, it permits the user to inspect the jobs on each queu to delete jobs from queues and to change the scheduled tim of execution for jobs on the queue of scheduled jobs. In t case of the module REG, it permits the user to add and dele clients from the list of registered client. In the case the module TARGET, it permits the user to inspect the jo running on each resource and also to change the thresho value for the maximum number of jobs which can be run on ea resource. In the case of the module ADB, it permits the us to inspect details of the jobs stored on both the databa and the files and also to change the selection procedure f transferring details of jobs from the database to the file

Although the transaction processing component 34 h been described with reference to a network manager, it can used in any computer system which has a client-serv architecture. Figure 7 shows a set of computers 50, 51, and 53 which are connected together by a telecommunicatic network 54. As indicated by the dotted line, the number computers connected in this way can be greater. Both clie and servers can be provided in any of the computers 50 to thereby providing the client-server architecture. In or to control the scheduling of requests to run jobs between clients and servers, one of the computers, for example computer 50, is provided with a transaction process

component generally similar to the transaction processing component 34 except that it does not include servers.

CLAIMS

1. A computer system having a client-server architecture said system comprising:

a set of clients;

a set of servers for serving requests from the client to run jobs;

means for managing requests from the clients to ru jobs on the servers; and

means for loading jobs on to the servers;

said request managing means being arranged to receiv requests from the clients and, on receiving a request, t perform the following operations:

check if a server is free to run the job;

if a server is not free, put the job on a queue fc jobs each of which is ready for execution when a serve becomes free; and

when a server is free to run a job, instructing sa: job loading means to load a job on to that server.

2. A computer system as claimed in claim 1, in which, tl request managing means is arranged to select the next job be loaded onto a free server in accordance with a predefin criterion.

3. A computer system as claimed in claim 1 or claim 2, which, on receiving a request to run a job, said reques managing means is arranged to perform the additiona following operations:

check if the job is scheduled for execution at a ti in the future; and

if the job is scheduled for execution at a time in t future, put a job on a queue for jobs each of which scheduled for execution at some time in the future.

4. A computer system as claimed in any one of t preceding claims, in which, on receiving a request to ru

job which is destined for a resource accessed by a server, said request managing means is arranged to perform the following additional operations:

check if the resource is free to run the job; and

if the resource is not free, put the job on a queue for jobs each of which is waiting for a resource to become free.

5. A computer system as claimed in claim 4, including a database for maintaining data on jobs running on the or each resource which is accessed by a server, for the or each resource said database keeping a count of the number of jobs being run on the resource and a threshold value for the number of jobs which can be run on that resource.

6. A computer system as claimed in claim 5, in which, said request managing means accesses said database in order to check if a resource is free to run on a job.

7. A computer system as claimed in claim 5 or claim 6, including means for permitting a user of the computer system to set the threshold value for the or each resource.

8. A computer system as claimed in any one of the preceding claims, further including means for creating and deleting servers, said server creating and deleting means being arranged to compare the number of jobs on the queue of jobs ready for execution with the number of servers for serving the jobs, and to create or delete servers in accordance with the result of the comparison.

9. A computer system having a client-server architecture, said computer system comprising:

a set of clients;

a set of servers for serving requests from the clients to run jobs;

means for managing requests from clients to run jobs in servers; and

means for loading jobs on to servers; said requests managing means being arranged to receive requests from the clients and, on receiving a request from a client to run job which is destined for a resource accessed by a server, to perform the following operations;

check is the resource is free to run the job; and

if the resource is not free, put the job on a queue for jobs each of which is waiting for a resource to become free.

10. A network manager for managing at least one element of a telecommunications network, said network manager including a computer system as claimed in any one of the preceding claims.

11. A method of operating a computer system having client-server architecture, said computer system comprising a set of clients and a set of servers for serving requests from the clients to run jobs, for each request made by a client to run a job on a server, said method comprising the steps of:

checking if a server is free to run the job; and

if a server is not free to run the job, putting the job on a queue for jobs each of which is ready for execution when a server becomes and

when a server is free to run the job, loading a job to the server.

12. A method of operating a computer system as claimed in claim 11, in which the method includes the step of selecting the next job to be loaded onto a free server in accordance with a predefined criterion.

13. A method of operating a computer system as claimed in claim 11 or claim 12 in which, for each request to run a job on a server, said method including the additional steps of:

checking if the job is scheduled for execution at a time in the future; and

if the job is scheduled for execution at a time in the future, putting the job on a queue for jobs each of which is scheduled for execution at a time in the future.

14. A method of operating a computer system as claimed in claim 11, claim 12 or claim 13 in which, for each request to run a job which is destined for a resource accessed by a server, the method includes the following additional steps:

checking if the resource is free to run the job; and

if the resource is not free, putting the job on a queue for jobs each of which is waiting for a resource to become free.

15. A method of operating a computer system having a client-server architecture, said computer system having a set of clients and a set of servers for serving requests from the clients to run jobs, for each request made by a client to run a job on a resource accessed by a server, said method including the steps of:

checking if the resource is free to run the job; and if the resource is not free, putting the job on a queue for jobs each of which is waiting for a resource to become free.

Fig.1.

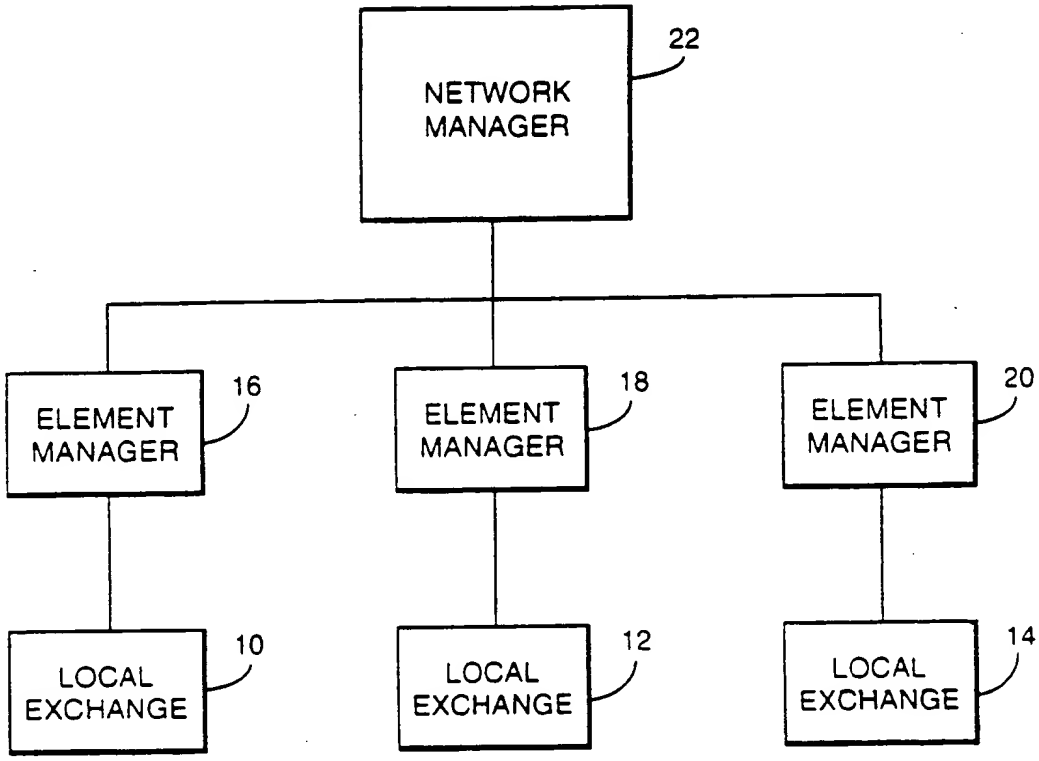


Fig.2.

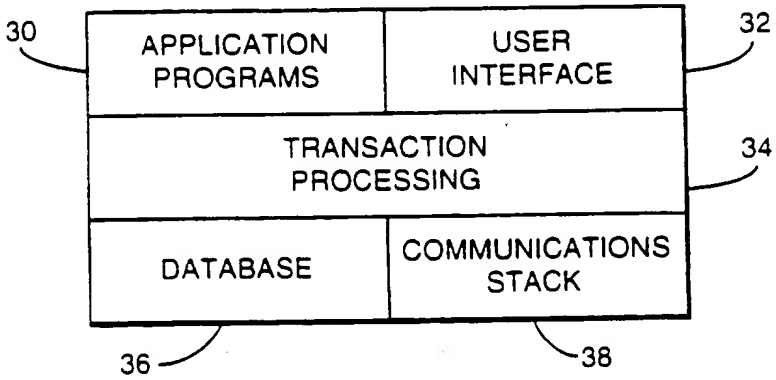


Fig.3.

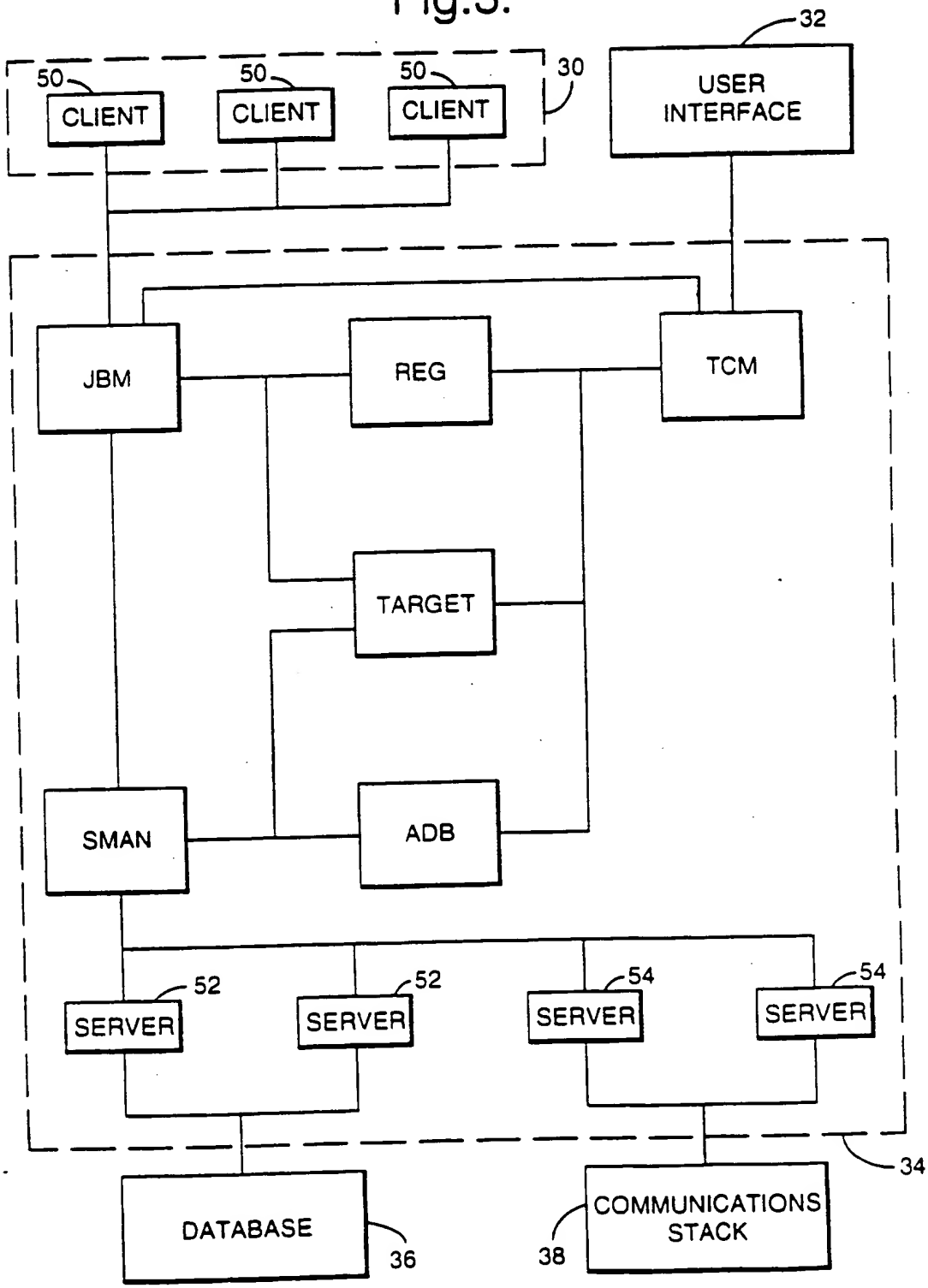

32
50
CLIENT
50
CLIENT
50
CLIENT
30
USER INTERFACE
JBM
REG
TCM
TARGET
SMAN
ADB
52
SERVER
52
SERVER
54
SERVER
54
SERVER
34
DATABASE
36
38
COMMUNICATIONS STACK

Fig.4.

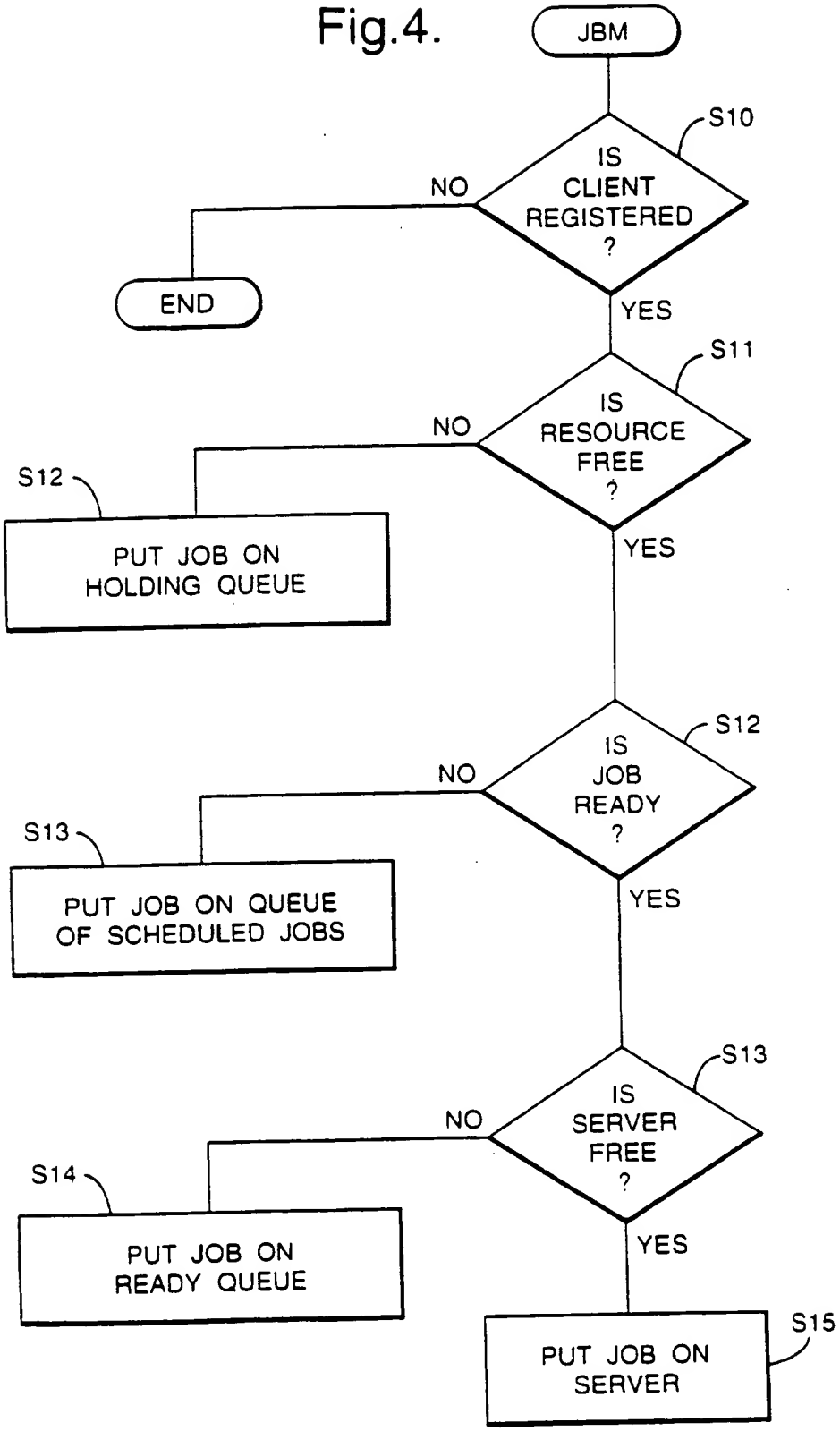

JBM
S10
IS CLIENT REGISTERED ?
NO
END
YES
S11
IS RESOURCE FREE ?
NO
S12
PUT JOB ON HOLDING QUEUE
YES
S12
IS JOB READY ?
NO
S13
PUT JOB ON QUEUE OF SCHEDULED JOBS
YES
S13
IS SERVER FREE ?
NO
S14
PUT JOB ON READY QUEUE
YES
S15
PUT JOB ON SERVER

Fig.5.

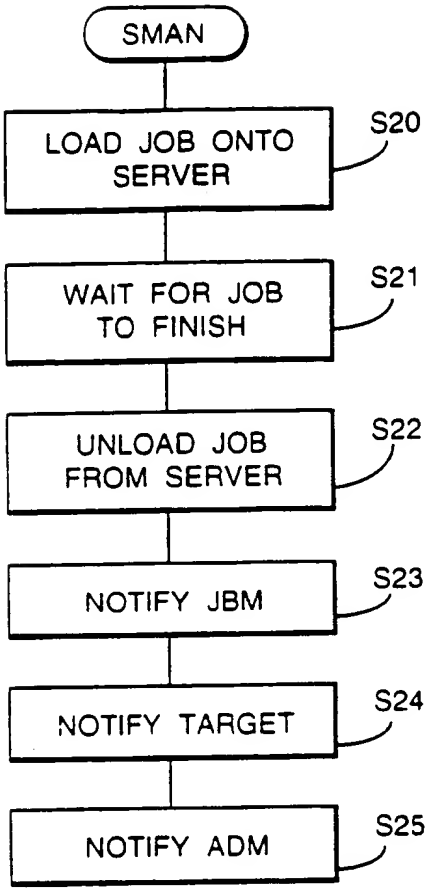

SMAN
LOAD JOB ONTO SERVER
S20
WAIT FOR JOB TO FINISH
S21
UNLOAD JOB FROM SERVER
S22
NOTIFY JBM
S23
NOTIFY TARGET
S24
NOTIFY ADM
S25

Fig.6.

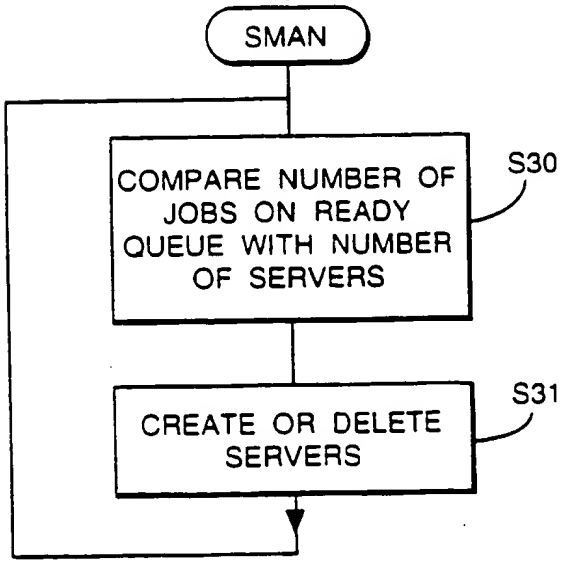
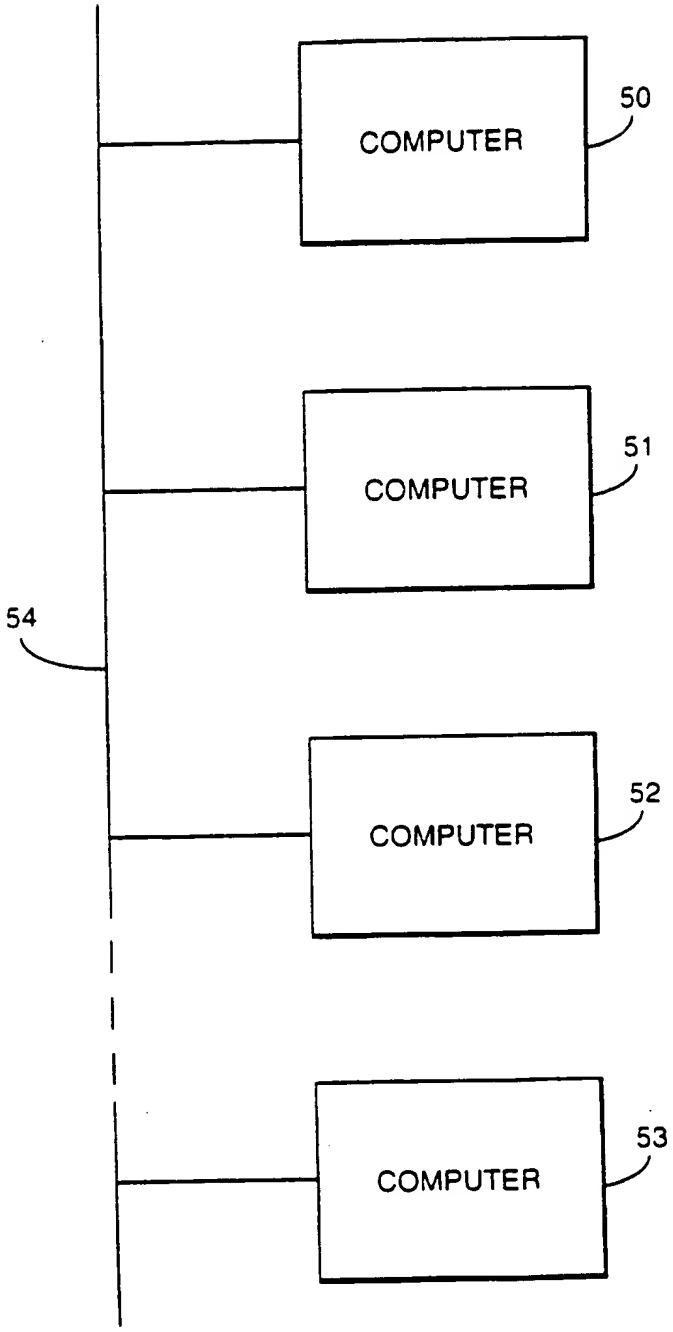

SMAN
COMPARE NUMBER OF JOBS ON READY QUEUE WITH NUMBER OF SERVERS
S30
CREATE OR DELETE SERVERS
S31

Fig.7.



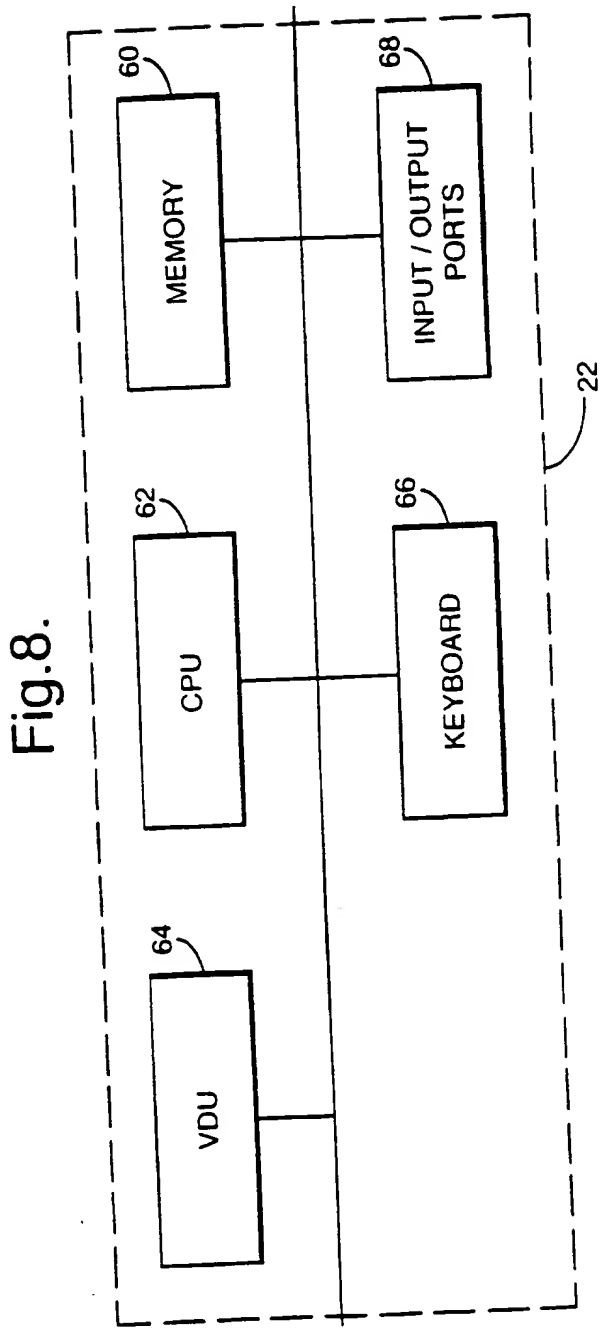


Fig.8.

INTERNATIONAL SEARCH REPORT

Interna . Application No
PCT/GB 95/01705

A. CLASSIFICATION OF SUBJECT MATTER
IPC 6 G06F9/46

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
IPC 6 G06F

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practical, search terms used)

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category *	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	US,A,5 249 290 (HEIZER) 28 September 1993 see the whole document ---	1-15
X	US,A,5 325 527 (CWIKOWSKI ET AL.) 28 June 1994 see the whole document ---	1-4,9-15 5-8
A	EP,A,0 601 579 (MITSUBISHI) 15 June 1994 see the whole document ---	1-15
A	EP,A,0 384 339 (DIGITAL) 29 August 1990 see abstract see column 1 - column 4 see column 12, line 13 - column 15, line 11; figures 1-7 see claims 1-28 --- -/--	1-15

[X] Further documents are listed in the continuation of box C.

[X] Patent family members are listed in annex.

* Special categories of cited documents :

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier document but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art.

"&" document member of the same patent family

1

Date of the actual completion of the international search	Date of mailing of the international search report
18 October 1995	10. 11. 95

Name and mailing address of the ISA	Authorized officer
European Patent Office, P.B. 5818 Patentlaan 2 NL - 2280 HV Rijswijk Tel. (+ 31-70) 340-2040, Tx. 31 651 epo nl, Fax (+ 31-70) 340-3016	Wiltink, J

Form PCT/ISA/210 (second sheet) (July 1992)

C.(Continuation) DOCUMENTS CONSIDERED TO BE RELEVANT

Category *	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	MICROPROCESSING AND MICROPROGRAMMING, vol. 12, no. 3/4, October 1983 , AMSTERDAM, NL, pages 159-166, A. CORRADI ET AL.: 'Using the iAPX-432 System as a Support for Chill Parallel Constructs' see page 160, left column, line 9 - page 161, left column, line 22 ---	1,2,4,9, 11,15
A	HEWLETT-PACKARD JOURNAL, vol. 41, no. 2, April 1990 , US, pages 54-59, K.S. KLEMBA ET AL.: 'HP OpenView Network Management Architecture' see figures 1,4 -----	10

1

INTERNATIONAL SEARCH REPORT

Information on patent family members

Intern. al Application No
PCT/GB 95/01705

Patent document cited in search report	Publication date	Patent family member(s)		Publication date
US-A-5249290	28-09-93	NONE		
US-A-5325527	28-06-94	JP-A-	6301619	28-10-94
EP-A-0601579	15-06-94	JP-A-	6282516	07-10-94
		GB-A-	2273378	15-06-94
EP-A-0384339	29-08-90	AU-B-	611605	13-06-91
		AU-B-	4996190	13-09-90
		AU-B-	630291	22-10-92
		AU-B-	7603391	15-08-91
		CA-A-	2010762	24-08-90
		JP-A-	3116262	17-05-91
		US-A-	5341477	23-08-94